

Atty. Docket No. MS171134.1

UTILITY-BASED ARCHIVING

by

Eric Horvitz

CERTIFICATION

I hereby certify that the attached patent application (along with any other paper referred to as being attached or enclosed) is being deposited with the United States Postal Service on this date June 28, 2001, in an envelope as "Express Mail Post Office to Addressee" Mailing Label Number EL798606480US addressed to the: Box Patent Application, Assistant Commissioner for Patents, Washington, D.C. 20231.

Himanshu S. Amin

(Typed or Printed Name of Person Mailing Paper)



(Signature of Person Mailing Paper)

Title: UTILITY-BASED ARCHIVING

5

Technical Field

The present invention relates generally to computer systems, and more particularly to a system and associated methodology adapted to infer how to store an item based upon a cost-benefit analysis related to maintaining the item in an active, efficiently active state.

10

Background of the Invention

Computer systems and related technologies have become a staple in many aspects of modern society. People have come to rely on these systems as a tool for use in both personal and professional lives. These technologies have, among other things, provided for increased communication and sharing of information among individuals and entities. For instance, computer systems and related technologies are currently used in conjunction with the Internet and local area networks to enable people to access, receive, generate and share unprecedented amounts of data (*e.g.*, documents, spread sheets, presentations, Internet files and email).

15

20

While individuals and society as a whole benefit from such free flow of information, there are costs associated with managing such data. Moreover, these costs generally grow as the volume of shared information increases. Accordingly, since more and more information continues to be created and circulated among a greater number of users, an important use of computer systems and related technologies is that of data management.

25

30

With respect to email messages, for example, an individual may be inundated with a large number of new email messages. As such, a user may be required to spend significant time and energy reviewing, responding to, organizing and/or sorting through these messages. Moreover, the number of email messages received is often inversely proportional to the amount of time available. For example, a manager who oversees many employees may have very little time to sort through, organize and/or respond to a significant amount of email messages. However, such an individual tends to fall within a

group that receives a disproportionately large number of email messages. Some of the manager's email messages may contain information that is of interest (*e.g.*, status reports for ongoing projects). As such, the manager may wish to review these messages one or more times. Other messages may, instead, contain irrelevant information (*e.g.*, unsolicited junk mail). In this case, the manager may wish to spend as little time as possible dealing with these messages and, therefore, would prefer that these messages be discarded immediately. Unfortunately, conventional systems generally do not provide for automatically discarding irrelevant messages and/or prioritizing messages based upon relevance – the messages are not sorted based upon their respective values and sizes. To date, systems and methodologies merely archive messages based upon chronology. Moreover, these systems usually require some type of consent before items can be discarded. As an example, a user is often asked whether it is appropriate to update a list of messages (*e.g.*, whether to archive one or more less relevant messages).

Since information will increasingly be propagated among individuals, managing of such information becomes more of an issue. While it may be desirable to indefinitely maintain all relevant items in an active state so that they can be quickly and easily accessed, technological realities limit the number of items that can be maintained in such a state. More particularly, computer systems have memory limitations. Fast memory that allows computer systems to maintain items in an active state is more limited than slower archival memory. Fast memory is, therefore, generally more expensive than archival memory, making it more costly to store an item in an active state. Moreover, sorting through vast amounts of items in an active state becomes very laborious (*e.g.*, some individuals have thousands of e-mails in an active state) – thus creating a need for streamlining of active items.

Summary of the Invention

The following presents a simplified summary of the invention in order to provide a basic understanding of some aspects of the invention. This summary is not an extensive overview of the invention. It is intended to neither identify key or critical elements of the invention nor delineate the scope of the invention. Its sole purpose is to present some

concepts of the invention in a simplified form as a prelude to the more detailed description that is presented later.

The present invention provides a system and methodology operable to infer or approximate what to do with an item. In one example, a value density is used for item ordering, which yields an approximation for choosing the best assortment of items of different sizes and values for storing in a limited space (*e.g.* limited memory). The value density is obtained *via* a cost-benefit analysis to infer whether a user would prefer to archive, store as active or discard an item (*e.g.*, e-mail, text, document, web page, image, audio). The invention provides for streamlining the number of items stored as active in order to facilitate ease of review, access and searching of items. In the cost-benefit analysis, overhead attributable to keeping the item active is compared to gains associated with having quick and easy access to the item. A value density is thereby obtained as a measurement of the worth of the item given its size and utility. The value density provides a basis for comparing one item to another such that a decision can be made as to which items to maintain in an active state. Given a finite amount of active space within which to maintain items, items having greater value densities can be actively retained. In this manner, inefficient utilization of active space can be mitigated.

The system and associated methodology of the present invention are adaptable to computer related applications including, but not limited to, email and document retention systems. Since there is a limited amount of active memory on computer systems, the cost of maintaining an item in an active state is measured in terms of the item's size. The benefit of keeping the item in an active state is measured in terms of a probabilistic determination that a user will access the item. The output of a probabilistic determination is a reflection of the relevance and utility of an item. In accordance with an aspect of the present invention, notions of temporal sensitivity of the likelihood that an item will be needed in the future can be leveraged, such that probabilistic determinations can continually be updated over time. As such, measurements of the utility of items and inferences drawn therefrom may be ongoing.

According to another aspect of the present invention, an item can be classified as either an item that will be accessed more than once or just a single time. An item that will be accessed only a single time is referred to as a "one-shot" item. A one-shot item

can be archived after it is viewed since it is not likely to be accessed again. For example, if the item happens to be a short email message (*e.g.* “see you later”) sent in response to user initiated dialog by an entity with which the user frequently corresponds, it may be regarded as a one-shot message and be discarded or archived after it is reviewed. An item
5 can be branded a one-shot item based upon its determined probability and/or value density. For instance, if a determined probability/value density is less than a threshold probability/value density, an item may be regarded as a one-shot item (*e.g.* probability item will be read more than once is less than 0.5). Alternatively, an item may be branded as a one-shot item if its determined probability/value density changes by more than a
10 certain amount within a given period of time.

According to still another aspect of the present invention, a learning system can act upon an inference system to adjust inferences made thereby. For instance, the learning system can modify the manner within which the inference system decides that an item is to be regarded as a one-shot item.

15 An interactive user interface (“UI”) is provided that allows a user to personalize how items are stored, including how probabilistic and inferential determinations are made.

To the accomplishment of the foregoing and related ends, certain illustrative aspects of the invention are described herein in connection with the following description and the annexed drawings. These aspects are indicative, however, of but a few of the various ways in which the principles of the invention may be employed and the present invention is intended to include all such aspects and their equivalents. Other advantages and novel features of the invention will become apparent from the following detailed description of the invention when considered in conjunction with the drawings.
20

Brief Description of the Drawings

Fig. 1 is a schematic diagram illustrating a system that provides for inferring whether to actively store an item in accordance with an aspect of the present invention;

Fig. 2 is a schematic diagram illustrating tables of information that may be stored in a property log in accordance with an aspect of the present invention;

Fig. 3 is an illustration of assortments of items stored in active and archive item stores at different times, namely t_1 , t_2 and t_3 in accordance with an aspect of the present invention;

Fig. 4 is a schematic diagram illustrating a system in a networked environment that provides for inferring whether to actively store items for multiple users in accordance with an aspect of the present invention;

Fig. 5 is a schematic diagram illustrating a system that provides for inferring whether to actively store items and to optimize usage of active space in accordance with an aspect of the present invention;

Fig. 6 is a schematic diagram illustrating a system that includes a learning component operable to adjust inferences regarding whether to actively store items in accordance with an aspect of the present invention;

Fig. 7 is an illustration of an interactive user interface (UI) in accordance with an aspect of the present invention;

Fig. 8 is a curve illustrating the probability that a user will access an item over time;

Fig. 9 is an alternative curve illustrating the probability that a user will access an item over time;

Fig. 10 is a flow diagram illustrating a methodology to infer whether to actively store an item in accordance with an aspect of the present invention; and

Fig. 11 is a schematic block diagram illustrating a suitable computing environment in accordance with an aspect of the present invention.

Detailed Description of the Invention

The present invention is now described with reference to the drawings, wherein like reference numerals are used to refer to like elements throughout. In the following description, for purposes of explanation, numerous specific details are set forth in order to provide a thorough understanding of the present invention. It may be evident, however, to one skilled in the art that the present invention may be practiced without these specific details. Moreover, well-known structures and devices are illustrated in some instances in block diagram form in order to facilitate description of the present invention.

The present invention relates to a system and associated methodology adapted to infer what to do with an item, and more particularly whether to archive and/or keep active an item based upon a cost-benefit analysis. The cost-benefit analysis determines the overhead associated with keeping the item active (*e.g.*, not archiving it) versus the gains in connection with having quick and easy access to the item. The cost of maintaining an item in an active state is measured in terms of the size of the item which, in turn, affects the amount of space needed to store it. The benefit of keeping the item active is measured in terms of a probabilistic determination that a user will access the item, which is a reflection of the relevance and utility of the item. The invention contemplates temporal sensitivity such that determined values and inferences can be dynamically updated over time.

Referring initially to Fig. 1, a system 100 is illustrated that provides for inferring whether to keep active, delete or archive items 102 numbered 1 through N (N being an integer). An inference system 110 receives the items 102 and makes an inference as to how a user would like to store the respective items 102. The inference system 110 may also receive and utilize extrinsic data 104 to decide how to store the items. An active item store 120 is employed to store items in an active state, while an archive item store 130 can be utilized to retain certain items in an archived state. Items in the active store are more readily available, while items relegated to the archive store are more difficult to access. For instance, an email message maintained in the active store may be presented to a user on a display (*e.g.*, monitor), such that the user can easily access it (*e.g.*, double clicking with a mouse). In the archive store, however, a user may be required to dig down through multiple archive files before being able to view a message, or only a link to

an item stored offline may appear. Although common reference is made herein to email messages, in accordance with the invention, items can be of any type susceptible to inference. For example, the items can be e-mail, documents, web pages, news articles, images or sound recordings. Examples of images include medical images such as microscope images, MRI images, X-rays, fingerprints, works of art, and videos, such as might be taken by a robot going about a task. Examples of sound recordings include music recordings and voice recordings.

The inference system 110 is operatively coupled to a property log 150 adapted to store information relating to items, users and extrinsic data. For email messages, for example, item information contained in the log 150 may include, among other things, data revealed within email headers, such as the title/subject matter of the message, context (*e.g.*, when/where/how/circumstances under which it was created), message priority (*e.g.*, whether it is urgent or of normal importance), the size of the message, whether there are any attachments and who sent the message (*e.g.*, an employer or an unknown entity that may send out unsolicited junk mail). User related information maintained in the log may include name, age, title, job description, current medium/environment being utilized (*e.g.*, office PC, car phone, personal digital assistant “PDA”), the time, date and frequency of access to one or more messages, and user preferences gathered from implicit evidence obtained by monitoring user activity. Extrinsic data in the log may include, but is not limited to, current date/time, news, headlines, holiday calendar dates, sporting events/announcements.

As shown in Fig. 2, some of this information may be stored in a tabular or grid-like format in the property log. More particularly, properties 210 numbered 1 through M can be maintained for items 202 numbered 1 through O (M and O being integers).

Similarly, properties 220 numbered 1 through P can be stored for users 230 numbered 1 through Q (P and Q being integers). As time goes on, entries within the log can be refreshed. For example, the number of times that a particular user accesses a certain item can be updated over time. Similarly, the medium that a user is engaged with can be refreshed over time (*e.g.*, where the user switches from his/her office PC to a car phone).

With reference back to Fig. 1, the inference system 110 is operable to utilize the information relating to items 102, users and/or extrinsic data 104 to infer how to store

items. The inference system does this, in part, by determining probabilities associated with items. More particularly, a probability component 112 of the inference system 110, calculates the likelihood that a user will access the items by applying the information through one or more probabilistic techniques. This can be expressed and read as:

$p(\text{access} | E)$; the probability that a user will access the item given some evidence.

The probabilistic techniques employed may include, but are not limited to, neural networks, naïve Bayesian processing, sophisticated Bayesian processing, similarity analysis employing dot product and/or cosine function processing and decision tree processing. By determining the likelihood that a user will access an item, the probability component 112 derives a metric indicative of the item's utility.

With continuing reference to Fig. 1, the probability component 112 is operatively coupled to a cost-benefit component 114 of the inference system. The cost-benefit component 114 is operative to calculate a value density based upon a determined probability and size of an item. As such, the value density is a measurement of the relative worth of an item given its size. More particularly, the value density measures the overhead attributed to keeping the item active versus the gains associated with having quick and easy access to the item. The value density can be expressed and read as:

$$\text{value density} = \frac{p(\text{access} | E)}{\text{item size}}; \text{ the ratio of probability of user access and item size.}$$

The inference system 110 can then utilize, among other things, determined probabilities and value densities to decide what to do with items or approximate orderings. By way of example, one or more rules can be utilized in the inference system. One such rule may indicate that an item should be moved into the archive item store 130 if its determined value density is below some threshold value. It is to be appreciated, however, that archiving rules can be applied to factors other than determined value

densities. The use of value densities for ordering is but one approximation for choosing the best arrangement of items to store in a limited space. The inference system is operative to utilize any suitable number of methodologies and/or models to infer what do with items. Additional criteria may also be utilized to control other aspects of items, such as the manner within which an item is presented. For instance, if user related properties reveal that the user is utilizing a PDA to access email messages, then the messages may be presented in a PDA ready format (*e.g.*, appropriately reduced resolution).

By way of further illustration, the inference system can be configured to make decisions as to whether an item is likely to be accessed only a single time and, if so, to brand that item as a “one-shot” item. To mitigate inefficient use of space, a one-shot item can be archived or discarded once it is accessed. In accordance with the present invention, the inference system 110 can examine items 102, extrinsic data 104 and/or information stored in the property log 150 to surmise whether an item should be regarded as a one-shot item. For example, the system may infer that a short reply message, such as “see you later” that does not include any attachments, is sent by someone with whom the user frequently corresponds and which is sent in response to user initiated dialog should be considered a one-shot message. Alternatively, the system may deduce that an email message which is sent from a user’s employer and which includes one or more attachments should not be regarded as a one-shot message.

Determined probabilities and/or value densities can be evaluated to gauge the status of an item. For instance, the results of a probabilistic determination can be compared to a threshold probability. If the determined probability is less than the threshold probability (meaning that it is rather unlikely that a user will access the item), then an inference can be made that the item is a one-shot item. Similarly, a determined value density can be compared to a threshold value density. If the determined value density is below the threshold value density, an inference can be made that the item is a one-shot item.

It is to be appreciated that temporal changes can be accounted for in the present invention. By leveraging notions of temporal sensitivity of the likelihood that an item will be needed in the future, determined values and inferences can be dynamically updated over time. For instance, the probability component 112 can continually

recalculate probabilities of items as new items become available and as information in the log 150 changes and is updated. The cost-benefit component 114 then utilizes current probabilities to determine contemporary value densities. Ongoing inferences can then be made about what to do with items. For instance, inferences can continually be made about whether items should be regarded as one-shot items. An item may, for example, be considered a one-shot item based upon a rate of decay or once its determined probability falls below some threshold probability within a predefined period of time (*e.g.*, five minutes). After an initial read, an email message may have a probability of being accessed again that, given the evidence and/or conditions, is adjusted downward to such a degree that it falls below some threshold value and should be archived after the initial read. Similarly, an item may be regarded as a one-shot item if its determined value density drops below some threshold probability within a predefined period of time. The process can start when a message is received wherein an initial probability of access is determined and then decays over time.

An exemplary effect of the system over time is illustrated in Fig. 3, wherein, at a first time t_1 , items 2, 3, 7, 8, 11 and 12 are stored in an active item store 320 and items 1, 4, 5, 6, 9 and 10 are maintained in an archived item store 330. At time t_2 , however, items have been rearranged based upon inferential determinations over time such that the active item store 320 contains items 1, 2, 4, 9, 10 and 11 and the archived item store 330 contains items 3, 5, 6, 7, 8 and 12. At time t_3 , the items have again been shuffled in accordance with temporally adjusted determinations such that items 1, 4, 7, 8, 9 and 10 are included in the active item store 320, while items 2, 3, 5, 6, 11 and 12 are positioned in the archived item store 330. It is to be appreciated that, in addition to storing items, the present invention contemplates discarding and/or recalling items.

With reference now to Fig. 4, it is to be appreciated that the present invention has application to networked environments wherein R number of users may receive and/or have access to S number of items 402 (R and S being integers). The inference system 410 can determine how individual users would like to store the items. In this manner, the number of items stored as active can be streamlined on a user by user basis such that ease of review, access and searching of documents is facilitated. In the example illustrated, for user 1, items 1, 4 through T are stored in an active item store 420 and items 2, 5 through

U are stored in an archived item store 430. In the same manner, for user 2, items 3, 4 through V are stored actively 422, while items 2, 7 through W are archived 432 and for user R, items 5, 6 through X are actively stored 424, while items 1, 6 through Y are archived 434 (T-Y being integers). It is to be appreciated that, while separate active and archived item stores are depicted, user based selections of items may be stored in a single active item store and a single archived item store (120, 130, Fig. 1). It is also to be appreciated that user specific selections can be updated over time. Although not shown, at a later point in time, for instance, the items may be rearranged on a user by user basis such that items 2, 7 through B, 4, 6 through C and 3, 4 through C are stored actively, while items 4, 5 through E, 5, 8 through F and 1, 9 through G are archived for users 1, 2, and R, respectively (B-G being integers).

With reference now to Fig. 5, in accordance with another aspect of the present invention, an optimization component 516 is included within an inference system 510. The optimization component 516 mitigates difficulties associated with storing a plurality of items 502 numbered 1 through H in a limited amount of active space (H being an integer). More particularly, where several items are important enough to be maintained in an active item store 520, the optimization component 516 is operable to decide the best or most useful assortment of items. In operation, the optimization component may employ the use of utility directed knapsack methodologies in its computations. Determined value densities and the amount of active space available are but some of the factors that may be considered by the optimization component. In particular, *value density* to do an ordering is but one approximation of the knapsack algorithm for choosing the best items of different sizes and values into a limited space (e.g., limited memory). Approximation has certain guarantees on optimality (e.g., value no less than .5 optimal), but improvements can be achieved with more computational resources, e.g., employ a complete search (intractable) or limited search (more tractable) to find a best fit. Furthermore, it should be appreciated that the general knapsack problem of putting in items of different values and sizes has not been applied to the challenges noted herein with respect to the subject invention. Accordingly, in the example illustrated in Fig. 5, both a cost-benefit component 514 and active item store 520 are coupled to the optimization component 516. The optimization component obtains valued densities from the cost-benefit component

514 and the amount of active space available from the active item store 520.

A probability component 512 is also shown in the inference system 510. The probability component 512 is operative to apply information about the items 502, users and extrinsic data 504 to probabilistic techniques to determine the likelihood that an item will be accessed. The probabilistic techniques employed may include, but are not limited to, neural networks, naïve Bayesian processing, sophisticated Bayesian processing, similarity analysis employing dot product and/or cosine function processing and decision tree processing. The probability component 512 may obtain some of this information from a property log 550 that stores such information. The cost-benefit component 514 utilizes determined probabilities from the probability component 512 and item size to calculate respective value densities of items.

The optimization component 516 determines if all of the items worth storing in the active space will actually fit within that space. Items typically warrant storage in active space if their value densities are above some predefined threshold. To determine if these items will fit within the active item store, the optimization component 516 is operative to manipulate metrics in any suitable manner, such as by adding their respective sizes. The optimization component 346 can then compare this sum to the size of the active item store. If all of the items will not fit within the active item store, the optimization component can assess which items to relegate to an archive item store or discard. The optimization component 516 can make this determination in any of a variety of ways. For example, the optimization component 516 may be adapted to find the arrangement of items that will maximize the total value density (*e.g.*, the assortment of items have value densities that, when summed, yield a maximum value). To effect this, the optimization component can begin storing the items that have the greatest value densities in the active space 520. This can be done in a sequential manner until the next item in queue will not fit within the active space. The remaining qualifying items can then be moved to the archive item store 530 or discarded.

To further mitigate the inefficient use of the active space, the optimization component can also examine the remaining items (*e.g.*, those items that have value densities above some predefined threshold, but are not stored in the active space). If any of these items will fit within the residual amount of active space, they can be stored

accordingly. In this manner, the number of actively stored items is streamlined to facilitate ease of review, access and searching of items.

In the example illustrated in Fig. 5, items numbered 2, 5, 6, 8, 9, 10 through J are shown in the active space 520 (J being an integer). Presumably, this is the collection that best mitigates the inefficient use of active space. For example, as discussed above, this may be the arrangement of items that, when summed, yields a maximum value density. This grouping may also include one or more remaining items that just fit within the residual active space. For instance, the sum of items 2, 5, 6, 8, 9, 10 may yield a maximum value density, while remaining item J may include a smaller item that just fits into the active item store. Items numbered 1, 3, 4, 7, 11, 12 through K are shown as being stored in the archived item store 530 (K being an integer). This set contains those items that are not useful enough (*e.g.*, do not have sufficient value densities) to be stored in active space. It may also include one or more items that warrant being stored in active space, but will not fit within the remaining active space. For instance, items 11 and 12 may have been the next items in queue, but had to be relegated to the archive.

It is to be appreciated that temporal changes can be accounted for such that the active space can be utilized efficiently over time. To do so, the probability component 512 continually updates or recalculates respective probabilities of items as new items are added and/or as information is updated in the property log. The cost-benefit component 514 can then utilize these probabilities to determine contemporary value densities. With updated value densities, the optimization component 516 can constantly re-arrange the items that are stored in the active space 520 and the archive 530.

Turning to Fig. 6, a learning component 660 may be operable to act on an inference system 610 and affect decisions made thereby. The learning system 660 can employ manual and/or automated means to analyze information relating to items 602, users and extrinsic data 612, some of which may be obtained from a property log 650. Conditional probabilities including, but not limited to, Bayesian statistical analysis can be utilized by the learning system. Results from computations performed by the learning system can be utilized to create and/or adapt decisions made by the inference system. The learning system 660 can operate continually such that one or more inferences about how to store items evolve over time. For example, the manner within which the inference

system 610 recognizes one-shot items can be updated as the learning system observes user activity and notes the items that are accessed only once and the circumstances surrounding such activity, including information stored in the log 650.

Turning now to Fig. 7, an interactive user interface (UI) 700 in accordance with an aspect of the present invention is illustrated. The UI 700 allows a user to personalize how items are stored. In the example shown, the UI is adapted to display L number of conditions 702 that affect how items are to be handled (L being an integer). Some of the entries are policies 710 that affect probabilistic determinations. Other entries are archiving rules 720. Still others 730, 740 regard item discard and memory utilization protocols, respectively. The UI is interactive in that it provides for user enablement/disablement and/or customization. More particularly, in the example shown, activation boxes are included. A user can click a mouse on these boxes to enable/disable the corresponding entries. For instance, activation box 752 next to the second policy 712 is checked. In this fashion, the second policy has been enabled and the probability will, therefore, be reduced by a certain amount after a user definable period of time passes during which the message is not accessed. Similarly, box 754 next to the second archiving rule 722 is checked. Accordingly, the second archiving rule has been enabled and therefore a message will only be archived once it exceeds a certain age. The UI also includes entry boxes that enable a user to configure entries. For instance, a user can type a value into entry box 762 to specify the degree to which the second policy should reduce the probability based upon the age of the message. The age itself is also user definable via box 764. Likewise, a user can type a value into box 766 to customize the second rule. It is to be appreciated that not all of the policies need to have the capability to be enabled/disabled and/or customized by a user. For instance, the third rule 724 shown in Fig. 7 is a firm rule that can neither be enabled/disabled nor customized. Although activation and entry boxes are illustrated, the present invention contemplates any suitable number and/or type of user interface elements (*e.g.*, slide bar elements, drop down menus, dials, buttons, speech input/output elements). Additionally, it is to be appreciated that any suitable number of entries can be incorporated into a UI, such as that shown in Fig. 7. For instance, there are different manners in which to specify goals of an archiving system - per user preferences, and thresholds, cost-benefit analyses, and specifications about free

space (e.g., “leave at least x megs of free space in the active store”, “archive any item with less than a $p(\text{access again} \mid E)$ of .05”, “archive any item with less than a value density of less than .1”).

Turning now to Fig. 8, since the present invention contemplates temporal sensitivity, a plot 800 of the probabilistic determination for an item is illustrated over time. In the graph, the probability that the item will be accessed given some evidence is plotted on the Y-axis and time is plotted on the X-axis. The probability curve is illustrated as decaying over time. The curve is drawn in this manner because empirical testing has provided support for the proposition that items become less useful over time and thus it is less likely that an item will be accessed as time goes on. Accordingly, at time T_0 , which is the first instance of the item, the probability that a user will access the item is at a maximum value, namely P_{\max} . At this point, the probability is, in all likelihood, high enough to warrant maintaining the item in an active state. As time goes on, however, the likelihood that the item will be accessed decreases. Thus, at some later point in time, namely T_{out} , the probability that a user will access the item decreases to a value, namely P_{out} , where it is no longer warranted to maintain the item in an active state. At P_{out} , the probability is at a level that, considering the size of the item, the value density is likely below some threshold level. At this point, an inference system may determine that the item no longer qualifies for storage in active space. In this fashion, the item can be aged out of an active item store. The curve shown in Fig. 8 schematically represents modeling $p(\text{access} > 1)$ where T_0 starts at the time a message is first read.

With reference now to Fig. 9, a slightly different probability versus time curve 900 is shown. While the curve generally shows a decaying probability over time, some notable changes in the probability are also reflected. The curve initially has a probability P_{init} at time T_{init} . The probability then drops rather suddenly from P_{init} to P_1 in a relatively short period of time, namely from T_{init} to T_1 . This may correspond to the situation where an email message is accessed and reviewed by a user shortly after it is received. The degree to which the probability drops is a function of the policies in place and information regarding the item, user and extrinsic data. For example, if the message is a short reply message, such as “see you later” which is sent by someone with whom the user frequently corresponds and which is sent in response to user initiated dialog, the

policies in place may cause the probability of that message to drop significantly after the message is accessed a first time. Alternatively, however, the situation is also illustrated in the curve wherein the probability increases slightly over time, namely from probability P1 at time T1 to probability P2 at time T2. This may, for example, occur where the email message is re-accessed several times by a user and the policies are configured to reflect the premise that where a message is re-accessed several times, the probability that the user will again re-access the message increases over time. This may be the case, for instance, where the email message contains a code and/or password. A user may repeatedly access the email message to retrieve the code and/or password until he/she has it memorized. After the user has the code and/or password memorized, the message is no longer re-accessed by the user, and therefore, the probability will be reduced over time. Thus, the probability subsequently falls to some value, namely Pthresh at time Tthresh, wherein it no longer warrants being maintained in active memory. At this point, the item can be aged out and archived or discarded. The curve shown in Fig. 9 could also apply to an item already in the archive whose probability of being accessed again is updated over time. For example, certain items--even those stored offline, might increase in p(access again) | E) where interest of user is learned with respect to a particular topic(s), via aged email. Updates can continually be reviewed to facilitate making decisions about whether to bring back items into the active store from the archive.

In view of the foregoing graphical, structural and functional features described above, a methodology in accordance with various aspects of the present invention will be better appreciated with reference to Fig. 10. While, for purposes of simplicity of explanation, the methodology of Fig. 10 is shown and described as occurring serially, it is to be understood and appreciated that the present invention is not limited by the illustrated order, as some aspects could, in accordance with the present invention, occur in different orders and/or concurrently with other aspects from that shown and described herein. Moreover, not all illustrated features may be required to implement a methodology in accordance with an aspect the present invention. It is further to be appreciated that the following methodology may be implemented as computer-executable instructions, such as software stored in a computer-readable medium. Alternatively, the methodology may be implemented as hardware or a combination of hardware and

software.

Fig. 10 illustrates a methodology 1000 for determining what to do with an item, and more particularly whether to archive or actively retain the item. The methodology begins at 1002 wherein general initializations occur. Such initializations can include, but are not limited to, allocating memory, establishing pointers, establishing data communications, acquiring resources, setting variables and displaying process activity. At 1004 a probability that the item will be accessed is determined based upon some evidence. The present invention contemplates that the evidence can include information about the item, user, and/or extrinsic data, where some of this information may be obtained from a property log. The probability that the item will be accessed is an indication of the relevance and utility of the item. After 1004, the methodology proceeds to 1006 wherein a value density is determined. The value density is a measurement of the worth of the item and may be determined by comparing the probability that the item will be accessed to the size of the item (which affects the amount of space the item will occupy). More particularly, the value density may be determined by finding the ratio of benefit of the item (measured in terms of the probability that the item will be accessed) to the cost of the item (measured in terms of the size of the item). After 1006, the methodology then proceeds to 1008 wherein an inference is drawn regarding how to store the item (*e.g.*, actively or not). This may include an evaluation of whether or not the item is a one-shot item and may be based upon conditions, some of which may be personalized. If the item is to be stored actively, the methodology proceeds to 1010 where the item is put through an optimization protocol to ascertain if the item should be stored in active space given the relative worth of the item as compared to other items and the amount of space available. A customized allocation of fast memory may be considered in 1010. If it is determined that the item is to be stored in active space, the methodology proceeds to 1012 wherein the item is stored in an active item store. If the determination in 1008 or 1010 is negative, the process proceeds to 1014 wherein the item is archived or discarded. Since the present invention contemplates temporal sensitivity, after 1012 and 1014 the method returns to 1002 so that the process can continually reassess whether the item is to be stored within active or archival space.

In order to provide additional context for various aspects of the present invention,

Fig. 11 and the following discussion are intended to provide a brief, general description of one possible suitable computing environment in which the various aspects of the present invention may be implemented. It is to be appreciated that the computing environment is but one possible computing environment and is not intended to limit the computing environments with which the present invention can be employed. While the invention has been described above in the general context of computer-executable instructions that may run on one or more computers, it is to be recognized that the invention also may be implemented in combination with other program modules and/or as a combination of hardware and software. Generally, program modules include routines, programs, components, data structures, etc. that perform particular tasks or implement particular abstract data types. Moreover, one will appreciate that the inventive methods may be practiced with other computer system configurations, including single-processor or multiprocessor computer systems, minicomputers, mainframe computers, as well as personal computers, hand-held computing devices, microprocessor-based or programmable consumer electronics, and the like, each of which may be operatively coupled to one or more associated devices. The illustrated aspects of the invention may also be practiced in distributed computing environments where certain tasks are performed by remote processing devices that are linked through a communications network. In a distributed computing environment, program modules may be located in both local and remote memory storage devices.

Fig. 11 illustrates one possible hardware configuration to support the systems and methods described herein. It is to be appreciated that although a standalone architecture is illustrated, that any suitable computing environment can be employed in accordance with the present invention. For example, computing architectures including, but not limited to, stand alone, multiprocessor, distributed, client/server, minicomputer, mainframe, supercomputer, digital and analog can be employed in accordance with the present invention.

With reference to Fig. 11, an exemplary system environment 1100 for implementing the various aspects of the invention includes a conventional computer 1102, including a processing unit 1104, a system memory 1106, and a system bus 1108 that couples various system components including the system memory to the processing

unit 1104. The processing unit 1104 may be any commercially available or proprietary processor. In addition, the processing unit may be implemented as multi-processor formed of more than one processor, such as may be connected in parallel.

5 The system bus 1108 may be any of several types of bus structure including a memory bus or memory controller, a peripheral bus, and a local bus using any of a variety of conventional bus architectures such as PCI, VESA, Microchannel, ISA, and EISA, to name a few. The system 1100 memory includes read only memory (ROM) 1110 and random access memory (RAM) 1112. A basic input/output system (BIOS), containing the basic routines that help to transfer information between elements within the computer 10 1102, such as during start-up, is stored in ROM 1110.

The computer 1102 also may include, for example, a hard disk drive 1114, a magnetic disk drive 1116, *e.g.*, to read from or write to a removable disk 1118, and an optical disk drive 1120, *e.g.*, for reading from or writing to a CD-ROM disk 1122 or other optical media. The hard disk drive 1114, magnetic disk drive 1116, and optical disk drive 15 1120 are connected to the system bus 1108 by a hard disk drive interface 1124, a magnetic disk drive interface 1126, and an optical drive interface 1128, respectively. The drives and their associated computer-readable media provide nonvolatile storage of data, data structures, computer-executable instructions, etc. for the computer 1102. Although the description of computer-readable media above refers to a hard disk, a removable magnetic disk and a CD, it should be appreciated by those skilled in the art that other types of media which are readable by a computer, such as magnetic cassettes, flash 20 memory cards, digital video disks, Bernoulli cartridges, and the like, may also be used in the exemplary operating environment 1100, and further that any such media may contain computer-executable instructions for performing the methods of the present invention and or may contain components that are to be installed in accordance with an aspect of the present invention. 25

A number of program modules may be stored in the drives and RAM 1112, including an operating system 1130, one or more application programs 1132, other program modules 1134, and program data 1136. The operating system 1130 may be any 30 suitable operating system or combination of operating systems.

A user may enter commands and information into the computer 1102 through one

or more user input devices, such as a keyboard 1138 and a pointing device (*e.g.*, a mouse 1140). Other input devices (not shown) may include a microphone, a joystick, a game pad, a satellite dish, a scanner, or the like. These and other input devices are often connected to the processing unit 1104 through a serial port interface 1142 that is coupled to the system bus 1108, but may be connected by other interfaces, such as a parallel port, a game port or a universal serial bus (USB). A monitor 1144 or other type of display device is also connected to the system bus 1108 *via* an interface, such as a video adapter 1146. In addition to the monitor 1144, the computer 1102 may include other peripheral output devices (not shown), such as speakers, printers, etc.

The computer 1102 may operate in a networked environment using logical connections to one or more remote computers 1160. The remote computer 1160 may be a workstation, a server computer, a router, a peer device or other common network node, and typically includes many or all of the elements described relative to the computer 1102, although, for purposes of brevity, only a memory storage device 1162 is illustrated in Fig. 5. The logical connections depicted in Fig. 11 may include a local area network (LAN) 1164 and a wide area network (WAN) 1166. Such networking environments are commonplace in offices, enterprise-wide computer networks, intranets and the Internet.

When used in a LAN networking environment, the computer 1102 is connected to the local network 1164 through a network interface or adapter 1168. When used in a WAN networking environment, the computer 1102 typically includes a modem 1170, or is connected to a communications server on the LAN, or has other means for establishing communications over the WAN 1166, such as the Internet. The modem 1170, which may be internal or external, is connected to the system bus 1108 *via* the serial port interface 1142. In a networked environment, program modules depicted relative to the computer 1102, or portions thereof, may be stored in the remote memory storage device 1162. It will be appreciated that the network connections shown are exemplary and other means of establishing a communications link between the computers 1102 and 1160 may be used.

In accordance with the practices of persons skilled in the art of computer programming, the present invention has been described with reference to acts and symbolic representations of operations that are performed by a computer, such as the computer 1102 or remote computer 1160, unless otherwise indicated. Such acts and

operations are sometimes referred to as being computer-executed. It will be appreciated that the acts and symbolically represented operations include the manipulation by the processing unit 1104 of electrical signals representing data bits which causes a resulting transformation or reduction of the electrical signal representation, and the maintenance of data bits at memory locations in the memory system (including the system memory 1106, hard drive 1114, floppy disks 1118, CD-ROM 1122, and shared storage system 1110) to thereby reconfigure or otherwise alter the computer system's operation, as well as other processing of signals. The memory locations where such data bits are maintained are physical locations that have particular electrical, magnetic, or optical properties corresponding to the data bits.

As used in this application, the term "component" is intended to refer to a computer-related entity, either hardware, a combination of hardware and software, software, or software in execution. For example, a component may be, but is not limited to, a process running on a processor, a processor, an object, an executable, a thread of execution, a program and a computer. By way of illustration, both an application running on a server and the server can be a component. Additionally, as used in this application, "system" is a structure comprising one or more modules. A "module" is a structure comprising computer hardware and/or software. For example, a module can be, but is not limited to, a computer readable memory encoded with software instructions or a computer configuration to carry out specified tasks. By way of illustration, both an application program stored in computer readable memory and a server on which the application runs can be module. Due to the nature of modules, multiple modules can be intermingled and are often not separated from one another. Systems can likewise be intermingled and inseparable. Likewise, it is to be appreciated that a module can be a software object.

It is to be appreciated that various aspects of the present invention may employ technologies associated with facilitating unconstrained optimization (*e.g.*, example, back-propagation, Bayesian, Fuzzy Set, Non Linear regression, or other neural network paradigms including mixture of experts, cerebellar model arithmetic computer (CMACS), Radial Basis Functions, directed search networks, and functional link nets) may be employed.

What has been described above includes exemplary implementations of the

present invention. It is, of course, not possible to describe every conceivable combination of components or methodologies for purposes of describing the present invention, but one of ordinary skill in the art will recognize that many further combinations and permutations of the present invention are possible. Accordingly, the present invention is intended to embrace all such alterations, modifications and variations that fall within the spirit and scope of the appended claims.